**ArgoCD**

* Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.
* It will synchronize the desired state(stored in git) with actual state(in K8s).
* ArgoCD ensure that the cluster matches it.

Why ArgoCD: Traditional cd tools push deployments manually or via pipeline but ArgoCD automates this by pulling desired manifests from git.

**Benefits**:

* Gitpos Mode: Single source of truth.
* Visibility: Web UI (Access) and uses CLI to access realtime application health,
* Multi-Cluster Support: Manage multiple clusters from one control plane.
* Rollbacks: Just revert git commit.

**ArgoCD Architecture:**

* 1) Api Server: It will handle cli/ui requests.
* 2)Repository Server: It will connect to git/helm repo and it will put manifests.
* 3) Application Controller: It will reconcile desired states.
* 4)Redis: Caching for memory performance.

**Workflow:**

Developer pushes the code/manifests to Git repo - ArogCd- K8s cluster(syncs it).

**Core Concepts:**

**1) A CRDA( Custom Resource Definition) what to deploy.**

**2) Project: Logical grouping of applications.**

**3) Sync: It will bring live cluster state in line with git.**

**4) Hooks: Presync, sync, postsync, hooks for automation.**

**5) Sync status: Synced and out of sync status check.**

**Deployment Strategies**:

* 1)Manual sync where admin triggers sync automatically.
* 2) Auto-sync: Argocd automatically applies changes from git.
* 3) Self-Heal: Automatic self heal.

Argo CD automates the deployment of the desired application states in the specified target environments. Application deployments can track updates to branches, tags, or be pinned to a specific version of manifests at a Git commit.

**Requirements**

* Installed [kubectl](https://kubernetes.io/docs/tasks/tools/install-kubectl/) command-line tool.
* Have a [kubeconfig](https://kubernetes.io/docs/tasks/access-application-cluster/configure-access-multiple-clusters/) file (default location is ~/.kube/config).
* CoreDNS. Can be enabled for microk8s by microk8s enable dns && microk8s stop && microk8s startS

**Steps:**

Step 1: Install ArogCD.

kubectl create namespace argocd

kubectl apply -n argocd -f <https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml>

This will create a new namespace, argocd, where Argo CD services and application resources will live.

Step2. Download ArogoCD cli:

curl -sSL -o argocd-linux-amd64 https://github.com/argoproj/argo-cd/releases/latest/download/argocd-linux-amd64

sudo install -m 555 argocd-linux-amd64 /usr/local/bin/argocd

rm argocd-linux-amd64

Step 3. **Access The Argo CD API Server**

By default, the Argo CD API server is not exposed with an external IP. To access the API server, choose one of the following techniques to expose the Argo CD API server:

Service Type Load Balancer

Change the argocd-server service type to LoadBalancer:

kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "LoadBalancer"}}'

Step 4.

Kubectl get all -n argocd: To access the Loadbalancer ip address to access argocd.

Step 5. Login Using The CLI

The initial password for the admin account is auto-generated and stored as clear text in the field password in a secret named argocd-initial-admin-secret in your Argo CD installation namespace. You can simply retrieve this password using the argocd CLI:

argocd admin initial-password -n argocd

Using the username admin and the password from above, login to Argo CD's IP or hostname:

argocd login <ARGOCD\_SERVER>

Change the password using the command:

argocd account update-password

Access the application using Loadbalancer Ip address: port number.

**To configure the application to automate the process.**

Left Panel- Settings -Select Repositories - Select + Connect Repo- Connection Method- HTTP/HTTPS.-Connect repo using HTTPS/HTTPS.

Select git- Choose project- default, Paste repo url with manifest files in a separate folder in root directory. Give github username and PAToken in the password section. Then select Connect at the top.

On left panel select Applications- New App- Give app name in small case letters.- Project-default-Sync Policy-Automatic and enable auto sync. Next Source give github url- Branch give as per your repo like main or master. Path: give folder path from github (ex.K8s-manifests) if manifests files are in this folder. Destination- Cluster url-Select Automatic from dropdown- Give the namespace as in our code. - At the top- Create.

We will get status and health monitoring details on the home page.   
If we make any changes in the github repo it will automatically be synced with the argocd and reflect in the app health section.  
We can also increase replica count and can verify it in our cluster using kubectl get all -n adi.

Configuration of Prometheus and Grafana can also be done using ArgoCD.\

